UNIP
UNIVERSIDADE PAULISTA

Linguagem de Programação Orientada a Objeto

Classes, Atributos e Métodos

Professora Sheila Cáceres

Orientação a objetos

- É uma forma de entender e representar sistemas complexos como estruturas hierárquicas de objetos que se relacionam.
- No enfoque de OO, as unidades básicas são os objetos que trocam mensagens entre si.
- Este modelo assemelha-se ao mundo real, onde todos os objetos tem atributos e atividades associadas a eles.
- Java é uma linguagem orientada a objetos pura, o que significa que tudo em Java é acessado através de classes e objetos. Diferente de linguagens híbridas tais como C++ e Object Pascal

Objetos e classes

- Objetos
  - Representam ‘coisas’ do mundo real ou do domínio de algum problema (exemplo: “o carro vermelho ali no estacionamento”).

Ferrari do Luciano

- Classes
  - Representam todos os tipos de objetos (exemplo: “Classe carro”).

Exemplo

Classe

Casa

Objetos

(Instancias da classe)

Casa do Presidente

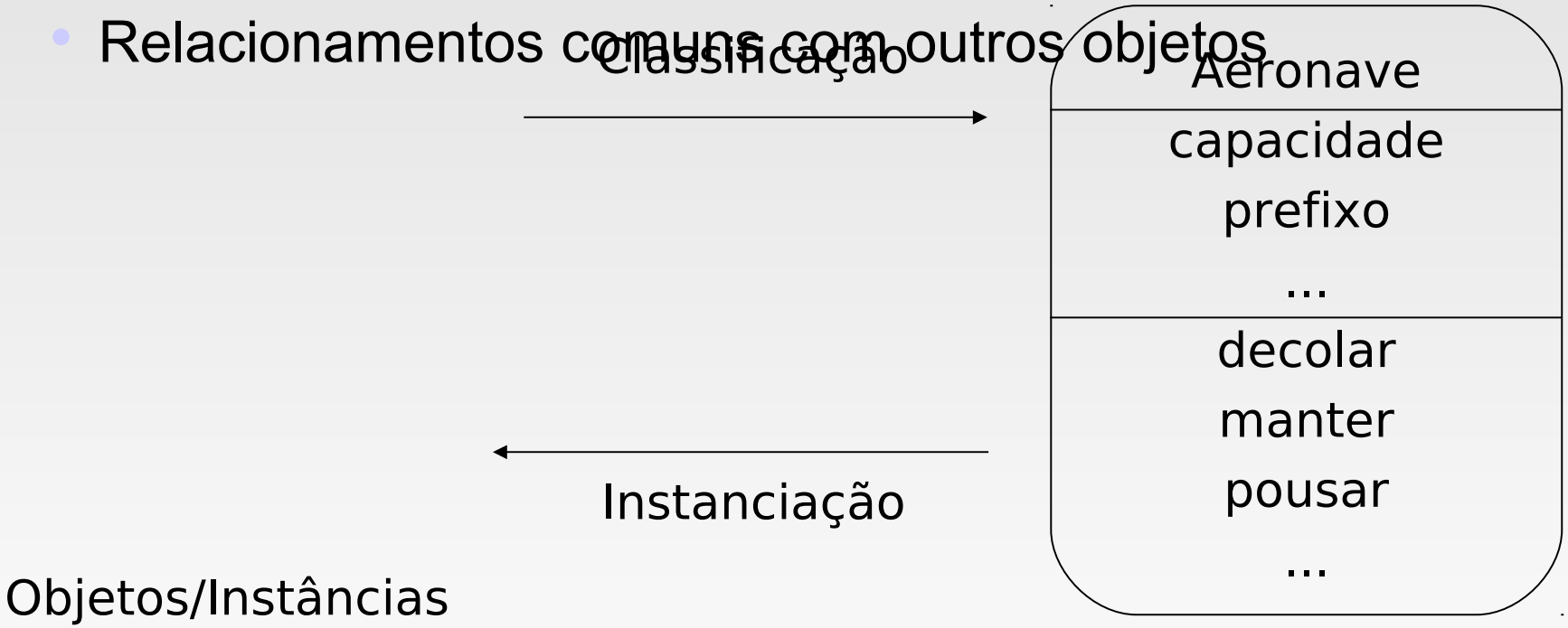
Casa da Maria

Casa Amarela

Objetos e Classes

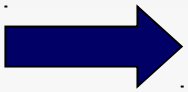
Classes

- Uma classe de objetos descreve um grupo de objetos com:
  - Propriedades (atributos) e comportamentos (métodos) semelhantes
  - Relacionamentos comuns com outros objetos



Classes

- Uma classe de objetos descreve um grupo de objetos com:
  - Propriedades e comportamentos semelhantes
  - Relacionamentos comuns com outros objetos
- Uma classe Java é um **molde para a criação de objetos**. A classe define as propriedades (atributos) e os comportamentos (métodos).
- Além disso, uma classe Java define como produzir (instanciar) objetos a partir dela.
- Para escrever uma classe (Aluno por exemplo), fazemos:



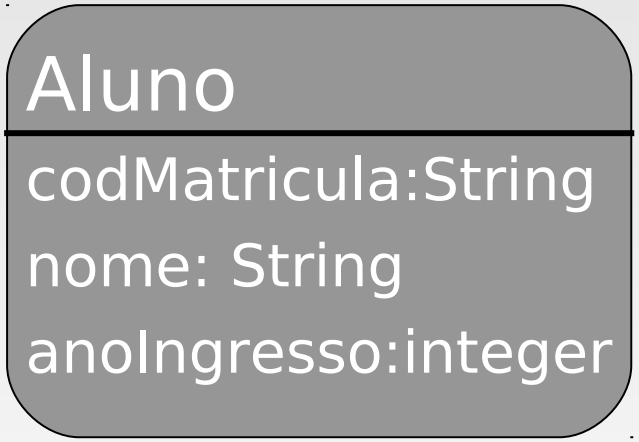
```
class Aluno{
}
```

Objeto

- É uma construção de software definida a partir de uma classe.
- São também chamados de **instâncias de classes.**
- Para uma mesma classe é possível definir mais de um objeto, cada um com suas características particulares.

Atributos

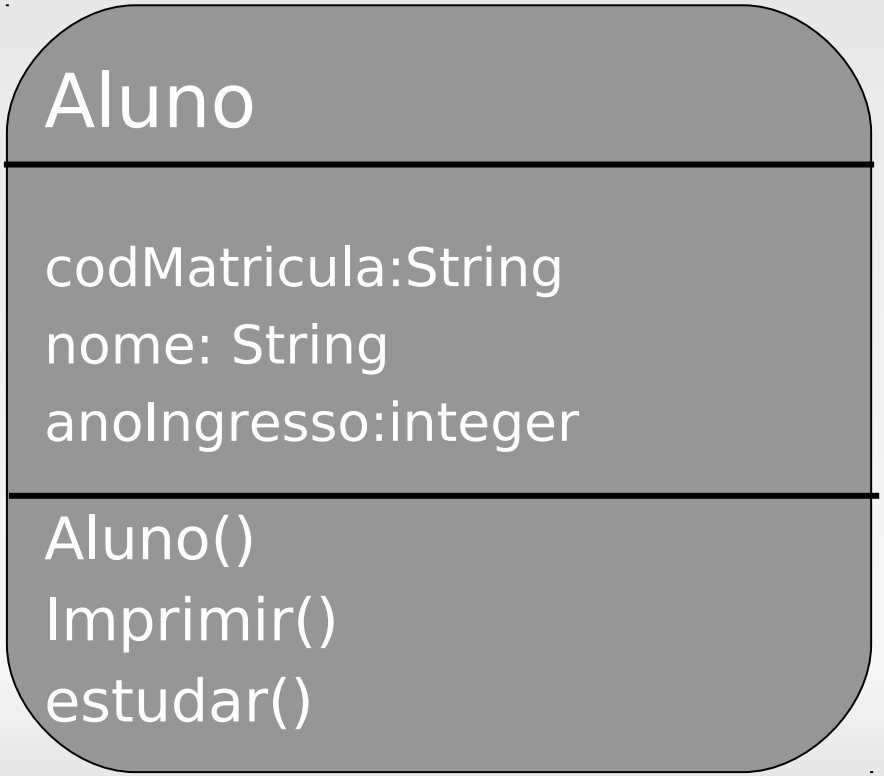
- São as características gerais definidas em uma classe, como por exemplo: cor, tamanho, nome, ordem, etc. São também chamados de **dados-membro**.




```
class Aluno{
    String codMatricula;
    String nome;
    int anoIngresso;
}
```

Métodos

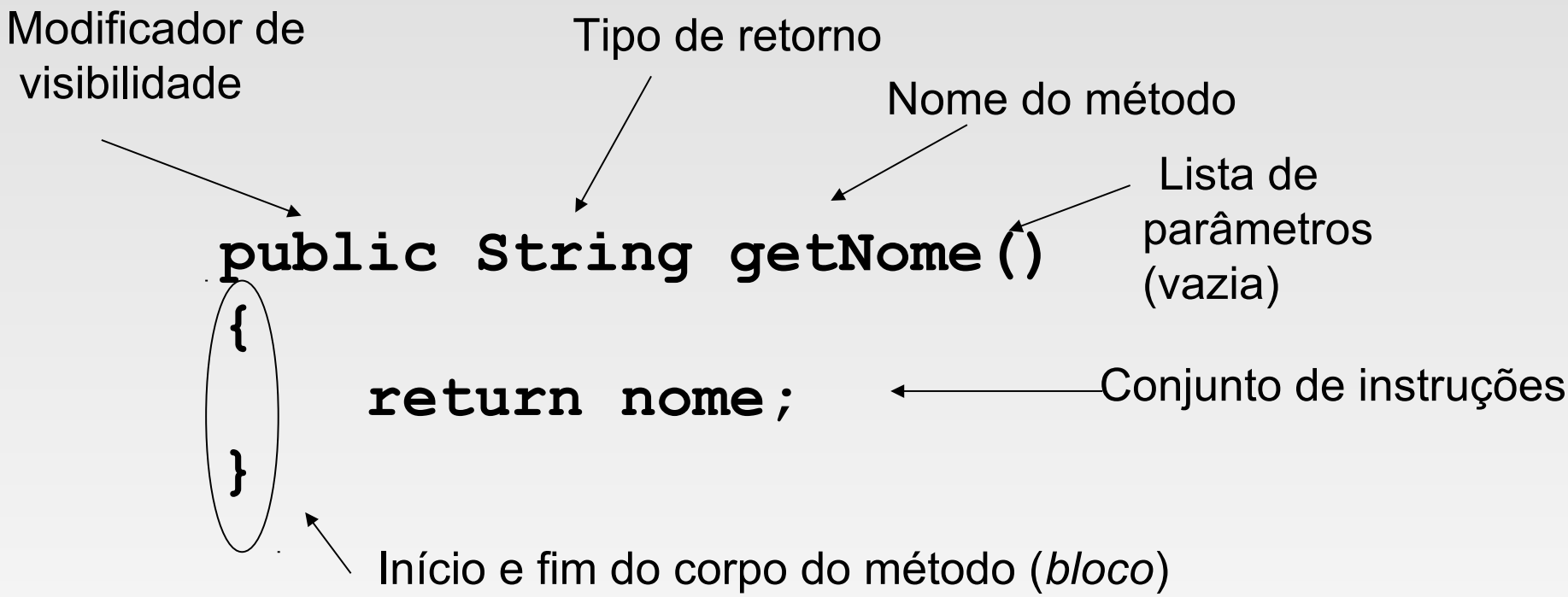
- São as atividades realizadas e podem usar os atributos da classe.
- Exemplo: na classeAluno, o método imprime() toma o conteúdo dos atributos e os exibe na tela.




```
class Aluno{
  String codMatricula;
  String nome;
  int anoIngresso;

 void imprimir()
 { System.out.print("Nome: "+nome);
 }
   void estudar(){
      System.out.print("Eu estudo");
   }
}
```

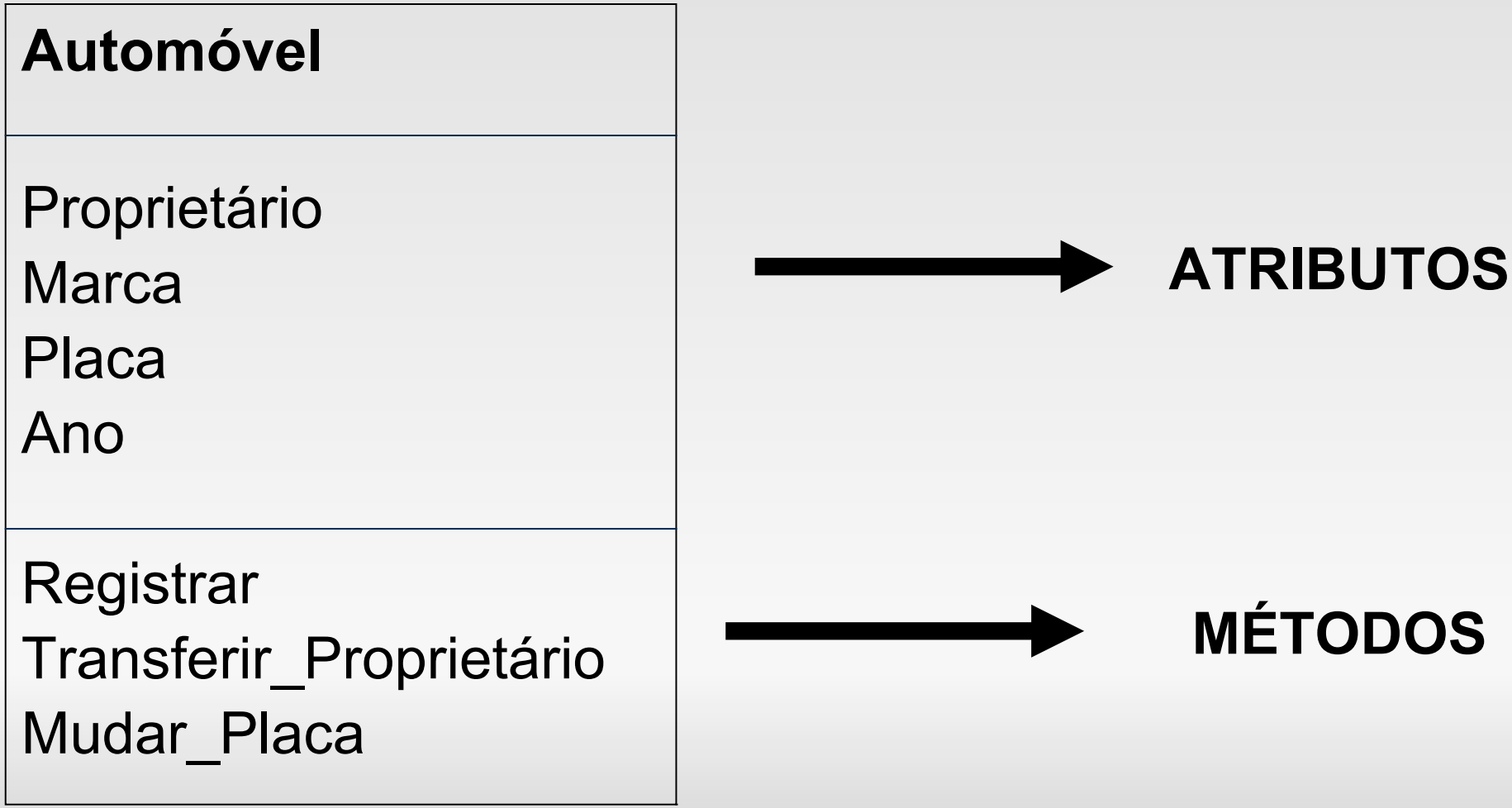
Métodos


Modificador de visibilidade
Tipo de retorno
Nome do método
Lista de parâmetros (vazia)
public String getNome()
{
return nome;
}
Conjunto de instruções
Início e fim do corpo do método (*bloco*)

Métodos

- Métodos têm uma estrutura que consiste em um cabeçalho e um corpo.
- O cabeçalho define a *assinatura do* método. `public int getNome()`
- O corpo engloba as instruções do método.
- Entre os métodos destacamos os métodos de acesso e os métodos modificadores
  - Os métodos de **acesso** fornecem informações sobre um objeto (**get**).
  - Os métodos **modificadores (set** por exemplo**)** modificam o estado de um objeto. São alcançados por meio da modificação do valor de um ou mais campos.
    - Geralmente contêm instruções de atribuição.
    - Geralmente recebem parâmetros.

Atributos e Métodos (Exemplo)



Método Construtor

- Servem para inicializar objetos
- Sempre têm o mesmo nome da classe
- Não pode especificar um valor de retorno
- Cuida da alocação de todos os recursos necessários para o objeto e retorna uma instância do objeto
- Podem ou não conter parâmetros
- Pode haver mais de um por classe (overloading - sobrecarga)

Por Exemplo: 

```
Aluno(String pMat,String pNome,
                        int pAnoIng)
  { codMatricula = pMat;
    nome = pNome;
    anoIngresso = pAnoIng;}
```

Instanciando una classe (Criando um objeto)

- 1° Criar uma referência para o objeto usando uma variável do tipo definido pela classe. Essa variável representa o nosso objeto.

```
Aluno al;
```

- 2° Deve-se alocar recursos para a referência criada. Esta alocação é feita pelo operador new

```
al = new Aluno("0001","alTeste01",1999);
```

- O operador new chama o construtor da Classe.para criar o objeto.
- Uma vez criado o objeto, os atributos e métodos da classe podem ser acessados usando o operador “.”

al.estudar()

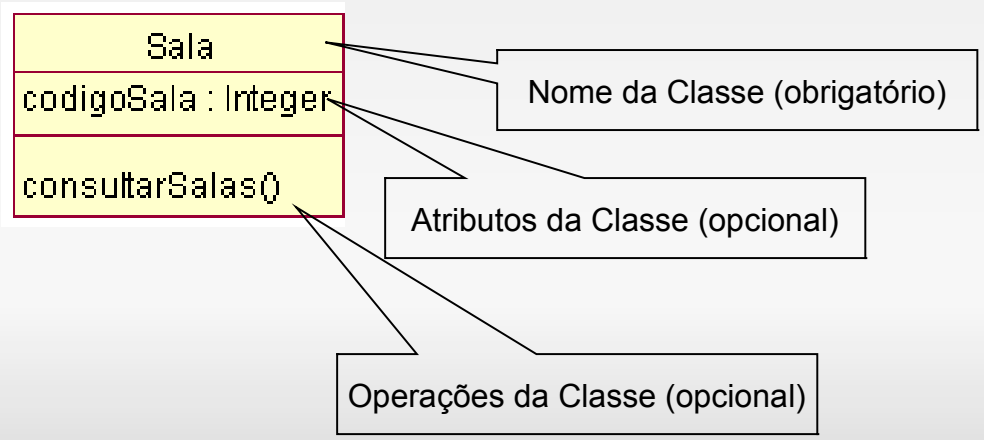
- Não se deve preocupar em destruir um objeto, isto é feito automaticamente pelo *garbage collector*

Diagrama de Classe

A UML é uma linguagem padrão para especificar, visualizar, documentar e construir artefatos de um sistema.

Em UML podemos representar um diagrama de classes (representação da estrutura e relações das classes que servem de modelo para objetos).

Uma classe em um diagrama de classes UML apresenta a seguinte representação gráfica:



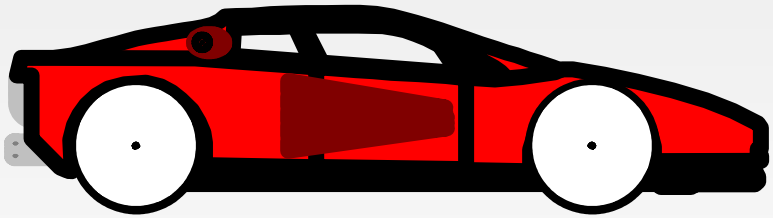
Implementação em Java

```
public class Sala{
   int codigoSala;

    public void consultarSalas(){...}
}
```

Estado do Objeto

- Conjunto de suas propriedades associadas a seus **valores**.
- Propriedades geralmente referenciadas como *Atributos*
- Ex:

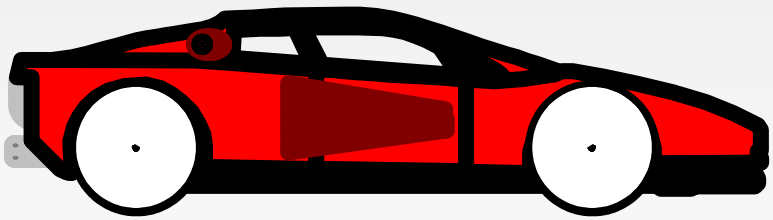


Atributos:

- Cor = **Vermelha**
- Ano = **2001**
- Velocidade = **0 Km/h**
- Combustível = **Gasolina**

Comportamento do Objeto

- Conjunto de serviços ou operações que outros objetos podem requisitar
- Operações geralmente referenciadas como *Métodos*
- Representa como o objeto reage às *Mensagens* a ele enviadas
- Ex



Métodos

- Acelerar ( )
- Frear ( )
- Acender Faróis ( )
- Virar a Direita ( )

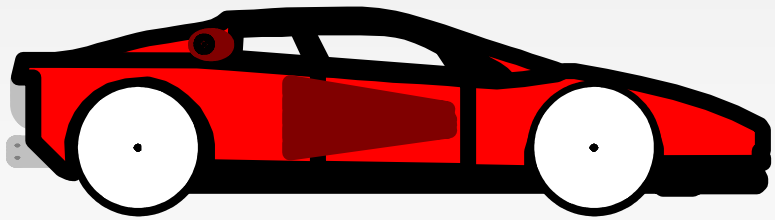
Cual seria o estado? E qual o comportamento das instancias da classe pessoa?

Pessoa
Nome CPF Idade Sexo
imprime() cadastra()

Objeto 01	**Nome**: Alexandre **CPF**: 111.254.648-77 **Idade**: 20 **Sexo**: M	imprime() cadastra()
Objeto 02	**Nome**: Maria **CPF**: 239.745.957-35 **Idade:** 36 **Sexo**: F	
Objeto 03	**Nome**: Juarez **CPF**: 001.895.457-47 **Idade:** 45 **Sexo**: F	

Identidade Única

- Objetos têm existência própria
- São distintos mesmo se seus estados e comportamento forem iguais
- Identificador que permite referência inequívoca
- Ex



Ferrari do Luciano

Referências

- Santos, Rafael; Introdução a Programação Orientada a Objetos Usando Java / Rafael Santos – Rio de Janeiro: Campus, 2003. Material no site editora.
- Deitel, H.M; Java Como Programar. Ed. Bookman, 2005.
- Programação Orientada a Objetos com Java, David J. Barnes and Michael Kolling. Pearson 2004.
- Material do professor Marco Fagundes, UFPa, 2003.

- Nota: O material da apresentação foi extraido de algumas das fontes aqui apresentadas